Extraction of CIM-Based Distribution Grid Topology Information for Observability

Shahid, Kamal; Schiavone, Enrico ; Drenjanac, Domagoj; Bæklund, Rasmus Pedersen; Olsen, Rasmus Løvenstein; Schwefel, Hans-Peter Christian

Published in:
15th European Dependable Computing Conference (EDCC)

DOI (link to publication from Publisher):
10.1109/EDCC.2019.00040

Publication date:
2019

Document Version
Accepted author manuscript, peer reviewed version

Extraction of CIM-Based Distribution Grid Topology Information for Observability

Kamal Shahid
Department of Electronic Systems,
Aalborg University, Denmark
Email: ksh@es.aau.dk

Enrico Schiavone
ResilTech S.R.L, Italy
Email: enrico.schiavone@resiltech.com

Domagoj Drenjanac
GridData GmbH, Germany
Email: drenjanac@griddata.eu

Rasmus Pedersen Bæklund
Thy-Mors Energi, Denmark
Email: rbl@thymors.dk

Rasmus L. Olsen
Department of Electronic Systems,
Aalborg University, Denmark
Email: rlo@es.aau.dk

Hans-Peter Schwefel
GridData GmbH, Germany
Email: schwefel@griddata.eu

***Abstract*—In order to implement fault-detection and diagnosis applications in Low Voltage (LV) grids, the data from customer connections needs to be processed jointly with measurements from the distribution grid by other Distribution System Operator (DSO) systems and in addition correlated to the LV grid topology. In practical DSO systems, the LV grid topology data is included in asset management databases and may use the Common Information Model (CIM) as data model. This grid topology information plays an important role in fault-detection and diagnosis. Thus, this paper presents an architecture and a concrete implementation to extract relevant grid topology information for use in fault detection and diagnosis from a CIM based asset management database. The approach is demonstrated and validated via CIM-based grid topology model from a real medium-sized distribution grid operator.**

***Keywords*—Low Voltage Grid Observability, CIM XML, Grid Topology Data, Smart Grid Data Integration**

I. Introduction

The introduction of renewable generation provides challenges to the power grids, specifically regarding power quality. On the other hand, photovoltaics (PV) and other grid-connected systems like storage also provide great opportunities. These opportunities include the provision of new measurement points, which to a large extent are already connected to Internet portals of the inverter vendors. The inverters are not only a source of grid-related measurements, but they can also provide actuation opportunities. A second opportunity is created by the massive deployment of Smart Meters (SM). Currently, SM measurements mostly target only billing and advanced customer information services, while the measurements from the SMs provide tremendous opportunities for the Distribution System Operator (DSO) to obtain valuable information about the Low Voltage (LV) grid.

Although an increasing number of grid-related data sources is in principle accessible to DSOs, this information is in most cases not yet utilized for detection, diagnosis, and localization of faults in the LV grid. In order to realize such applications in LV grids, the data from the SMs, Inverters, and other measurement devices of the DSO needs to be correlated to the LV grid topology. Fusing heterogeneous measurements to DSO data (a.k.a. Smart Grid Data Integration) provides multiple challenges, such as [1]: (1) Measurements from devices with diverse identification schemes need to be related to a unique grid topology model. (2) Algorithmic challenges to deal with varying data quality as well as potentially inaccurate data. (3) The interconnection of different systems with different criticality levels. Therefore, both the algorithmic solutions as well as the connectivity and security from an ICT perspective need to be designed carefully.

In order to store information about grid components, some DSOs use the CIM - IEC 61970, which is a well-known industry standard data model, to describe networked physical assets in an electrical power grid [2]. CIM based data models are standard in commercially available SCADA systems and are used for a very detailed description of grid assets. However, currently these have limited application in power system analysis mainly due to the complexities associated to grid topology processing and thus not employed for grid observability applications. This calls for a need of efficient grid topology processing mechanisms i.e. extracting relevant topology information and filtering out the relevant information that are based on existing power industry data standards such as CIM and IEC-61850 [3] etc. needed for fault-detection and diagnosis applications in LV grids. There are different approaches for fault-detection, diagnosis and localization, see [4]. Active approaches and approaches that require highly synchronized high-resolution measurements, as e.g. provided by phasor measurement units, are however not cost-efficient for deployment in LV grids. Focus of the grid topology processing is therefore to obtain a grid representation that allows to correctly associate measurement data to the LV grid and to do a steady-state calculation of electrical parameters in the grid under different fault-assumptions.

The challenge of making data accessible from different subsystems and then to enable data-intensive services on top of such heterogeneous data has been previously investigated in the context of electrical power grids in [2], [5]–[8]: [2]

investigates the use of CIM as one of the data ingestion sources developing an interoperable power grid data model to satisfy grid analytic requirements. Reference [2] presents the work of abstraction and translation from CIM to a Bus-Branch model as a part of an overall system design and implementation for grid analytics and data interoperability. Reference [9] describes the usage and implementation of CIM for grid data model as the reference data model. Reference [8] explores the problem of translating data in the CIM XML format to the required format for such legacy power system analysis applications and present solutions to some of the challenges in data translation.

Specifically, the question addressed in this paper is: How to efficiently process very large CIM-XML based electrical power grid topology information to reduce the complexity of the data, making it easily usable for fault-detection and diagnosis in LV grids. The CIM-XML based electrical power grid topology data used in this work is based on a real multi-substation model of a distribution network.

The remainder of this paper is organized as follows: Section II provides the architecture of the data fusion solution for grid observability and specifically describes the so-called grid topology subsystem and introduces the Grid Topology Head-End. Section III describes the real-life CIM model that is currently being used by medium-sized DSO. Section IV describes the traversal algorithm for deriving network information from CIM-XML model that is deployed in the Grid Topolgy HeadEnd. Section V validates the algorithm and provides a preliminary performance analysis of the traversal algorithm. Finally, Section VI summarizes the paper and provides an outlook.

II. Framework Architecture for Processing Heterogeneous Data

This section introduces the overall architecture and the application context for the use of grid topology information.

A. General data processing architecture

Fusing data from different heterogeneous data sources in LV grids can add value to the DSOs in terms of increased grid observability required for fault detection and diagnosis applications. Moreover, the target of data fusion solution is to use off-the-shelf computing hardware and existing communication technologies to leverage measurement functionality. The solution correlates SM and Inverter measurements with information from existing DSO subsystems, in order to enable and develop novel LV grid observability applications for voltage quality, grid operation efficiency, and LV grid outage diagnosis. The achieved observability can subsequently be used by novel control coordination approaches, which may use the inverter actuation capabilities in conjunction with selected existing DSO actuation for voltage quality enhancement and loss minimization in the LV grid. The proposed system architecture is shown in Figure 1.

The proposed system architecture is very similar to a classical middleware-based architecture. The middleware layer,

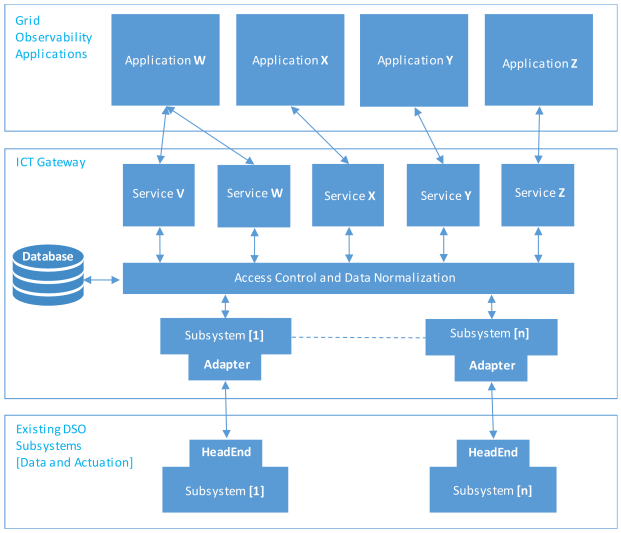

Fig. 1: System architecture for data fusion, adapted from [1]

here implemented by the ICT Gateway, provides a uniform application platform for domain-oriented applications by abstracting specific details about provider subsystem interface into a normalized/harmonized data model and by taking care of issues related to reliability and security. Note that the detailed description of complete system architecture and the interaction of different subsystems is out of the scope of this paper and can be found in [1], [10].

B. Topology data models for fault-detection and diagnosis

Grid observability applications are shown on the top of Figure 1: they use the measurements and the grid topology information in order to calculate and visualize derived metrics such as occurrence and location probability of faults. When using the architecture introduced in this paper, the applications can be implemented independently of the used subsystems as they only interact with measurement and actuation subsystems via the ICT Gateway. These applications may use an electrical model of the relevant parts of the distribution grid for the following purposes: (1) to enable to work with incomplete measurement scenarios, e.g. when not all customer connections are measured by a Smart Meter or an Inverter. (2) to calculate missing measurands, e.g. some measurement devices may not provide voltage measurements, but only active and reactive power. (3) To detect and discard or replace erroneous measurements. (4) To calculate electrical parameters resulting from faults assumed in different locations. Given the expected data quality and measurement resolution in LV grids, transient calculations of the LV grid are out of scope and only steady-state grid calculation e.g. using Power-Flow calculations [11] complemented by pseudo-measurement generation [12] appears feasible; other examples of such grid model implementations may quantify the uncertainty of calculated electrical parameters [13].

In order to enable processing of measurements and calculation of missing electrical variables from a grid model, a representation of the grid topology is required. The latter is in essence an annotated graph, in which edges represent cables or overhead lines, and nodes represent substation trafo, busbars in substation or junction boxes, sleeves or connected customers. Cables or overhead lines are annotated by relevant parameters, including in particular impedance and the presence of fuses. Nodes may have multiple attributes including type and Geolocation. The ICT Gateway then needs to link measurement of voltages to nodes and measurement of currents to cables or overhead lines. It is important to note that although many LV grids in principle show a meshed structure, their actual operation is frequently done in a fully radial structure. The underlying structure is then used for manual opening and closing of disconnectors in order to form different radial topologies, see also [14]. As the topology processing in this paper derives the operational electrical grid topology, this resulting topology is in the following assumed to be radial, i.e. form a tree. Note that topology changes via such manual disconnection occur on timescales of at least weeks to months within a single LV grid area.

C. Grid Topology Subsystem

The ICT Gateway will need to obtain the annotated graph representation of the LV grid topology, which was introduced in the previous subsection. In order to do so, it connects to one of the subsystems responsible for processing and keeping track of the grid topology. The HeadEnd of topology subsystem, called Grid Topology HeadEnd (GT-HE), sends processed topology information on request to the adapter of the same subsystem (see Figure 1). The design of GT-HE is the main contribution of this paper.

The GT-HE needs to implement the following functions: (1) It is an initiator of communication with the ICT Gateway over the corresponding adapter deployed on the ICT Gateway. The HE server thus performs authentication and registration with the adapter. (2) It can in a pull or push based manner provide updates of the current grid topology, i.e. list of nodes and connections in a specified topology, detailed information for a specified node or a connection. In order to provide these data, the GT-HE needs to connect to relevant IT subsystems at the DSO. In some DSOs, the relevant information is distributed over different IT systems, e.g. cable and nodes are in the GIS system while information about connected customers such as Meter IDs and peak power are in the customer information data base. The GT-HE then needs to obtain the data from the relevant IT systems and extract the required annotated grid topology graph.

In the following, we present the scenario of a real medium-sized DSO with around $50,000$ customers that stores its grid topology in a CIM based DB.

III. Practical CIM Example from Medium Sized DSO

This section introduces the entities captured in grid topology subsystem utilized by a DSO in one of the Scandinavian countries that reflects and describes the MV grid topology down to the LV grid topology. Moreover, the relation and function of these entities in the representative grid is also described in this section. It is important to note that the extraction of only a single LV grid area (behind one secondary substation) out of the complete distribution grid is addressed in this work. However, the grid topology subsystem has been designed to extract all LV grid areas available within the distribution grid. In the following we study a case of a topology which includes nearly $50,000$ households.

In the available topology data, a $Substation$ can refer to any of the three entities in the grid i.e. $PrimarySubstation$, $SecondarySubstation$ and $CableBox$. The entity called $PSRType$ determines the sub-type to which a $Substation$ refers.

The top-level entity in an LV grid is a Substation of $PSRtype\ SecondarySubstation$, which is an entry point from the medium voltage grid to the LV grid. This entity is connected to the MV grid on one side and to the LV grid on the other side. The $BusbarSection$ entity plays the same role as junction box: it has one cable as an input, but can have multiple cables as output via $ConnectivityNode$. An entity of type $ACLineSegment$ (cable) is responsible for making connections between all other entities via $ConnectivityNodes$ and $Terminals$. $ConnectivityNode$ represents points where terminals of AC conducting equipment are connected with zero impedance.

In order to model the details of a household connection and associate it with a metering device that acts as a data source, two entities are introduced i.e. $EnergyConsumer$ and $UsagePoint$. Here, $EnergyConsumer$ is a point in the network e.g. an end of house connection cable, while $UsagePoint$ is a logical or physical point in the network to which readings or events may be attributed. It is used at the place where a physical or virtual meter may be located. The $EnergyConsumer$ entity is connected to the $BusbarSection$ via $ACLineSegment$. To establish a relation between an $ACLineSegment$ (cable) and other entities, in particular when the entity is a household, the GT-HE has to be carefully designed and implemented.

IV. Processing CIM Data in GT-HE

Several modelling and mapping issues were identified and addressed while deriving network information from CIM-XML data by GT-HE. For instance, neither the provided data nor the CIM-standard indicates how the hierarchy of different entities should be built i.e. no explicit mapping between an $ACLineSegment$ and an $EnergyConsumer$ etc. An $ACLineSegment$ only refers to the entity/node (referred as $EquipmentContainer$) to which it is directly connected. There is no information regarding house number or even the

street and consequently, it cannot be uniquely mapped to a household. Similar is the case with other entities. Thus, due to the composite design pattern used throughout the data, entities cannot be directly interlinked.

In order to validate the output of the topology parser, a subset of topology data was extracted manually out of the data available in the CIM-XML topology file and converted to a diagram. Figure 2 depicts a small portion from the extracted LV grid topology. The figure shows how a Substation ($PSRType$ called $CableBox$) is connected to the $BusbarSection$, $Disconnector$, $ACLineSegments$ down to the $EnergyConsumer$ at the end. At several points, the topology structure can be seen to grow via $ConnectivityNodes$ and $Terminals$ (the whole extracted topology figure is intentionally not included due to the limited length of the paper).

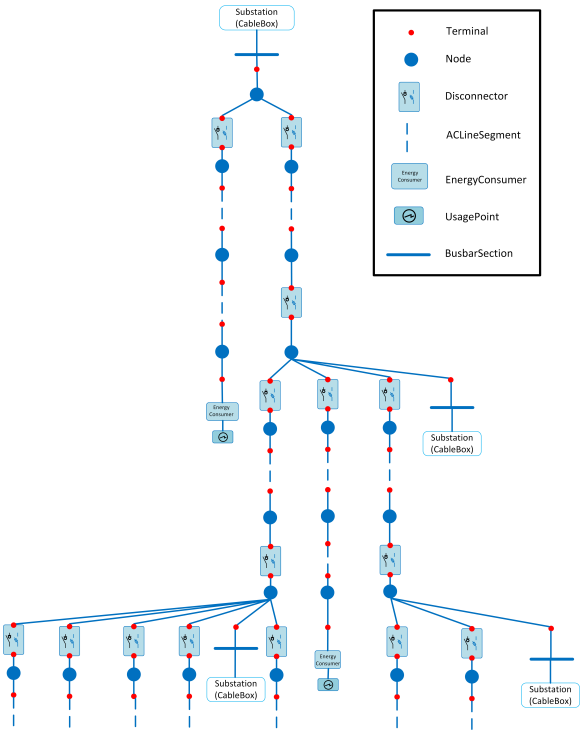

Fig. 2: Subset of LV grid topology extracted by GT-HE out of the complete CIM-XML based topology file.

A. Extraction of CIM-Based Power System Network Topology Model

As discussed in Section III, due to the lack of an explicit mapping between an $ACLineSegment$ and $EnergyConsumer$ etc., the main challenge is to find out a way to parse the whole file such that it helps creating a topology out of the available data. Thus, it is necessary to start scanning from an entity e.g. a Substation and find out the adjacent entities via associated attributes. This process continues until all entities are associated to their adjacent entity. Figure 3 illustrates the work-flow that the GT-HE has to conform with.

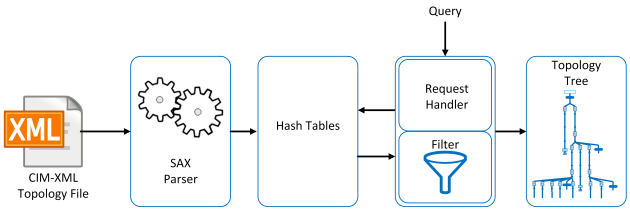

Fig. 3: Grid topology HE design.

XML Parser: The GT-HE starts by extracting and parsing substation data, which is then later referenced by subsequent entities. Then, it looks for the entities that refer the parsed $Substation$ as their equipment container and finds $BusbarSection$ as well as a $PowerTransformer$ (in case of a $SecondarySubstation$). The parser proceeds towards the last entity in the topology tree i.e. $EnergyConsumers$ by tracking the subsequent entities (such as $Terminals$, $ConnectivityNodes$, $Disconnectors$ etc.) and relating those to the already processed $Substation$ as well as the $BusbarSection$. The outcome from the GT-HE comprises different tables that reflect different node types and a table with the list of all cables where for each cable a starting entity and an ending entity are defined. It is important to note that the extracted entities and cables are provided on the interface towards the ICT Gateway. Data can either be pushed on each detected change, i.e., a topology update, in the subsystem or ICT Gateway can request the entities on-demand. Moreover, the subsystem can either provide the whole topology, a subset of nodes and related cables, or just give a detailed information about a particular node.

Java provides various ways to parse an XML file, such as, parsing an XML file using Document Object Model (DOM) parser, Simple API for XML (SAX) parser or Streaming API for XML (StAX) parser. For parsing the topology file, SAX parser was selected, because it parses the XML file line by line and it triggers events when it encounters an opening tag, a closing tag or character data in the file (thus it is also called an event-based parser) without loading the complete file in memory. Therefore, a SAX parser is usually used when processing very large XML documents whose object-model tree would consume too much memory. Although the SAX parser provides good CPU and memory efficiency, it possesses two disadvantages, i.e. [15]: 1) No random access to an XML document, since it is processed in a forward-only manner, and 2) If one needs to keep track of data that the parser has seen or where the parser has changed the order of items, then one must write the code and store the data on its own.

Thus, in the current implementation, the topology parser is designed such that it first reads the CIM-based topology file (XML format) via the SAX parser and then converts it into a set of hash tables. Additional methods allow traversing through these hash tables to create a data tree structure. This in turn leads to an easier and simpler interaction with the topology data. A challenge is that a child reference is only provided

one way as shown in below XML example, i.e. it is simple to go from $VoltageLevel$ to $Substation$, but not the other direction.

As the parser reads through the complete file and being event driven, the steps needed to parse the file grow linearly with the number of CIM nodes, i.e. $O(n)$.

Hash Tables: The Hash tables in this context are used to map mRIDs to the CIM object instances. This is happening during the parsing phase of the CIM file. As we create one object instance for each CIM node, this part is rather memory consuming, but if fitting in RAM of the host computer, it ensures constant lookup time for each CIM node in the tree, which later is critical for fast service to topology dependent applications. Due to the previous described internal referencing, each hash table index leads to a vector of elements that refers to that particular mRID. Storing a vector of referring mRID's for each mRID index in the hash table, allows us to solve the issue of multiple references to a single element, and handle the indirect referencing at the same time.

Obviously, hash collisions should be avoided at all cost in this process, but as the provided mRID is 32 characters versions of a $128bit$ key, we see this as insignificant.

Request Handler: To make sense of the hash tables, we need a request handler, which is capable of tracing through the direct and indirect links in the CIM node object instances. The $getNode()$ and $getNodes()$ take an mRID as input to fetch a given $CIMNode$ from the Hash tables.

So in essence, the algorithm starts from an initial node and looks through the hash tables for direct references (the node's children) and lists of mRID's of other elements that points to the current node, put these on a stack and add them to a node tree. $BoundaryNodes$ are defined in a given query, e.g. "Secondary Substation", in which the algorithm checks if a node is of that type before it adds children to the stack. Another type of boundary node are equipment that have switch capability. Some of these are kept open normally, to assure the radial operation structure of the grid, while allowing redundant connections in case of failures and repair. A boundary has been reached if such node has been found and is open. Other checks also need to be done, and all are done via the function *isThisABoundaryNode(mRID)*. As the algorithm traverses the tree, it will run out of nodes, and the stopping condition will be met when all trees in the output tree has been visited, and the algorithm is guaranteed to terminate. Loops are avoided via the hashtable *doNotRevisit*.

Filter: The grid topology parser extracts all tags in the XML document, where only partial sets are needed. Thus the full set of tags needs to be broken into smaller and more manageable sets. Very few assumptions can be made to the topology, as grid topologies can vary from wide, balanced trees, to narrow unbalanced trees depending on the location (geographical and grid), which also makes it challenging to develop efficient tree parsers. The following algorithm recursively calls itself to filter the tree from previous step, and adds an additional $O(N \cdot Log(N))$ steps, where N is the number of elements in the topology tree.

Algorithm 1 Pseudo code for topology tree construction

1: **Step 0:** Initialize
- Vector resultTree<CimNode>,
- HashMap doNotRevisit <String, Boolean>,
- Boolean stoppingConditionIsTrue=false;
- Boolean boundaryNode = false;
- String mRID = getFirstNode();

2: **Step 1:** Traverse the tree
3: *Repeat*
4: clear stack;
5: CIMNode currentNode = getNode(mRID);
6: boundaryNode = isThisABoundaryNode(mRID);
7: **if** (CurrentNode has children & !doNotRevisit & !boundaryNode) **then**
8: For all direct referring, put child mRID on stack;
9: For all indirect referring nodes, put child mRID on stack;
10: **if** stack not empty **then**
11: For each stack item:
12: **if** !doNotRevisit(mRID) **then**
13: add to resultTree(mRID, idcounter); add to doNotRevisit(mRID);
14: increase idcounter;
15: **if** more nodes to assess **then**
16: mRID = next node in resultTree();
17: **else**
18: stoppingConditionIsTrue = true;
19: Until stoppingConditionIsTrue

Algorithm 2 Pseudo code for TraverseTree(CIMNode startingNode, int parentNode)

1: **Step 0:** Initialize
2: **Step 1:** Check if node is matching filter condition (type, attributes etc.)
3: **Step 2:** If match,
- create new node; add to outputtree;
- establish parent/child relation for parent and child

4: **Step 3:** check children of this node
5: **if** Children present **then**
6: **if** relevant node **then**
7: TraverseTree(nextNodeInTree, thisNodeID)
8: **else**
9: TraverseTree(nextNodeInTree, parentNodeID)

V. Validation and preliminary performance analysis

Although topology information does not change frequently, it is still essential also to assure that ICT Gateway has the

correct (updated) topology information. Therefore, the fault detection application should be able to obtain a local subset of the grid topology reactively from the information source i.e. GIS system just after the fault has occurred. For this purpose we seek to assess the time behaviour of the algorithm. The base distribution grid has roughly $50,000$ customers and a total of roughly $1.2mio.$ CIM elements in a $0.5GB$ CIM formatted XML file. An initial parsing of this file is done via a SAX parser, which takes around 10 seconds and uses around 800 MB of memory as the parser uses a number of hash tables for fast indexing. This is a one time operation only. From the full data set, around 330 subsets of smaller grids are randomly selected and extracted with boundaries to the nearest secondary substation, each containing 100-4000 CIM objects for 100-230 household per subset. A post filtering is applied to include most relevant CIM object (cables, substations, fuses etc.) reduces the amount of CIM object down to 50-1500 per subset. The process of creating a sub-tree based on the hash table indexes, is measured on an Intel Xeon, Windows 10 machine with 16GB of memory, and the result CDF for the 330 reduced subsets is shown in Figure 4.

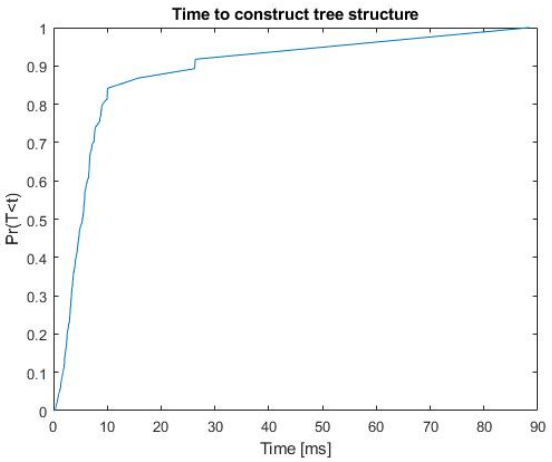

Fig. 4: Time to construct topology tree out of CIM-XML topology data.

The results show that the algorithm performs well in time at the cost of memory, and the topology trees are extracted extremely fast as compared to [2] where the algorithm was taking around 45 seconds to parse the data from a 30,000 node customer CIM data on an Intel®core i7 processor with 16 GB RAM. Moreover, the complexity of algorithm in [2] was $O(n^2)$, while the current algorithm uses $O(n \cdot log\, n)$ via hash maps. Since fault-localization in LV grids is expected to work on time-scales of several minutes, the resulting performance is sufficient.

VI. Conclusions and Outlook

This paper presents the architecture and a concrete implementation to extract relevant LV grid topology information for use in LV grid fault detection and diagnosis applications from a CIM based asset management database. The approach is demonstrated and validated on the CIM-XML model from a real medium-sized distribution grid operator. The challenge addressed in this paper mainly focused on efficiently processing CIM-XML based electrical power grid topology information to reduce the complexity of the data and making it easily usable for the grid observability applications. Future work should develop approaches to handle incomplete and erroneous topology data, investigating scenarios where topology data is distributed over different IT systems, and investigating strategies for updates of topology data. Moreover, in order to make the described process more efficient, the authors are considering approaches to employ database technology for persistence, indexing and efficient retrieval of CIM-based network models.

Acknowledgment

The research leading to these results has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 774145 for the Net2DG Project (www.net2dg.eu).

References

[1] H.-P. Schwefel *et al.*, "Net2dg deliverable d1.2 - initial baseline architecture," Tech. Rep., 2018.

[2] S. Shukla *et al.*, "Efficient method for extracting network information from the cim asset model," in *2016 IEEE Power Energy Society Innovative Smart Grid Technologies Conference (ISGT)*, Sep. 2016, pp. 1–5.

[3] R. E. Mackiewicz, "Overview of iec 61850 and benefits," in *2006 IEEE PES Power Systems Conference and Exposition*, Oct 2006, pp. 623–630.

[4] A. Bahmanyar *et al.*, "A comparison framework for distribution system outage and fault location methods," *Electric Power Systems Research*, vol. 145, pp. 19–34, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0378779616305302

[5] Y. Pradeep *et al.*, "Cim-based connectivity model for bus-branch topology extraction and exchange," *IEEE Transactions on Smart Grid*, vol. 2, no. 2, pp. 244–253, June 2011.

[6] D. S. Popovic *et al.*, "Extension of the common information model with a catalog of topologies," *IEEE Transactions on Power Systems*, vol. 22, no. 2, pp. 770–777, May 2007.

[7] R. Ramanathan and B. Tuck, "Contingency analysis using node/breaker model for operation studies," in *2015 IEEE Power Energy Society General Meeting*, July 2015, pp. 1–5.

[8] A. McMorran *et al.*, "Translating cim xml power system data to a proprietary format for system simulation," in *IEEE Power Engineering Society General Meeting, 2004.*, June 2004, pp. 116 Vol.1–.

[9] J. A. R. Mondejar *et al.*, "Upgrid project - d2.4 report on the implementation of the cim as the reference data model for the project v1.2," Tech. Rep., 2016.

[10] H. P. Schwefel *et al.*, "Net2dg deliverable d3.1 - ict analysis and gateway design," Tech. Rep., 2018.

[11] R. M. Ciric *et al.*, "Power flow in four-wire distribution networks-general approach," *IEEE Transactions on Power Systems*, vol. 18, no. 4, pp. 1283–1290, 2003.

[12] K. Dehghanpour *et al.*, "A survey on state estimation techniques and challenges in smart distribution systems," *IEEE Transactions on Smart Grid*, vol. 10, no. 2, pp. 2312–2322, 2019.

[13] H. Schwefel *et al.*, "Using smart meter measurements to manage accuracy of current calculations in lv feeders," in *to appear in IEEE SmartGridComm*. IEEE, 2019.

[14] W. Pacific Northwest National Laboratory, Richland, "Grid archtecture, release 2.3," U.S. Department of Energy, Tech. Rep., 2014.

[15] "Java sax parser overview," https://www.tutorialspoint.com/java_xml/java_sax_parser.htm, (Accessed on 05/04/2019).